Ahmat Barkai Moussa

Task 9 - Excel Export ve Import : Oluşturduğunuz tablo için Backoffice üzerinden Excel ile veri import etme ve export etme işlemlerini gerçekleştirin. Import sırasında veri mevcut ise güncellesin, mevcut değil ise yeni eklesin. Mevcut olup olmadığının kontrolü için unique bir alana bakılmalıdır. (Task 5'de yaptığınız gibi tek bir alan index değilken iki alan bir araya gelince unique oluyor, query'de buna göre yazılmalıdır.) Referans için ReferenceDistance tablosu için yazılan import ve export kodlarına bakabilirsiniz.

1 – Oncelikle ExcelConstant'ta TrainingProduct interface oluşturuyoruz ve attributleri yazıyoruz.

```
interface TrainingProduct{

    String PRODUCTNAME ="Product Name";

    String COMPANYNAME = "Company Name";


}
```

2- Tabloyu import etmek için ReferenceDistanceImportPerformer reference olarak alarak TrainingProductTableImportPerformer sınıfını oluşturuyoruz ve içinde gereken logic'lerimizi yazıyoruz .

```java
public class TrainingProductTableImportPerformer extends  AbstractMatchableExcelOperationPerformer{
    @Resource(name="trainingProductDao")
    private TrainingProductDao trainingProductDao;

    @Override
    protected void performInternal(Sheet sheet, BiFunction<Row, String, Optional<Cell>> cellIndexMap) {
        forEachRow(sheet, row -> getOrCreateTableRow(headerKey -> cellIndexMap.apply(row, headerKey)));
        getModelService().saveAll();
    }

    private TrainingProductModel getOrCreateTableRow(Function<String, Optional<Cell>> cellIndexMap) {
        String productName = getFullStringOrNull(cellIndexMap.apply(ExcelConstants.TrainingProduct.PRODUCTNAME).map(this::getCellAsString).orElse( other: null));
        String companyName = getFullStringOrNull(cellIndexMap.apply(ExcelConstants.TrainingProduct.COMPANYNAME).map(this::getCellAsString).orElse( other: null));

        final TrainingProductModel trainingProductModel =  trainingProductDao.getTrainingProductByProductNameAndCompanyName(productName,companyName);

        if(Objects.isNull(trainingProductModel)){
            return createModel(cellIndexMap);
        } else {
            if(StringUtils.isNoneBlank(productName)){
                trainingProductModel.setProductName(productName);
            }
            if(StringUtils.isNoneBlank(companyName)){
                trainingProductModel.setCompanyName(companyName);
            }
            return trainingProductModel;
        }
    }
    private TrainingProductModel createModel(Function<String, Optional<Cell>> cellIndexMap){
        TrainingProductModel trainingProductModel = getModelService().create(TrainingProductModel.class);

        trainingProductModel.setProductName(cellIndexMap.apply(ExcelConstants.TrainingProduct.PRODUCTNAME).map(this::getCellAsString).orElse( other: null));
        trainingProductModel.setCompanyName(cellIndexMap.apply(ExcelConstants.TrainingProduct.COMPANYNAME).map(this::getCellAsString).orElse( other: null));

        return trainingProductModel;
```

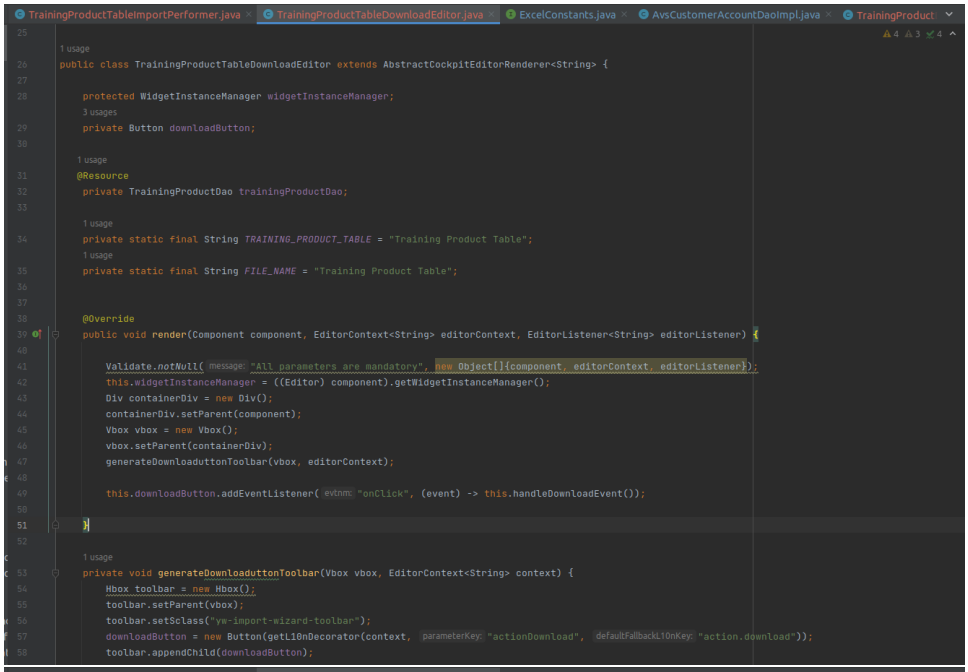
TrainingProductImportPerformer beans'leri yazıyoruz

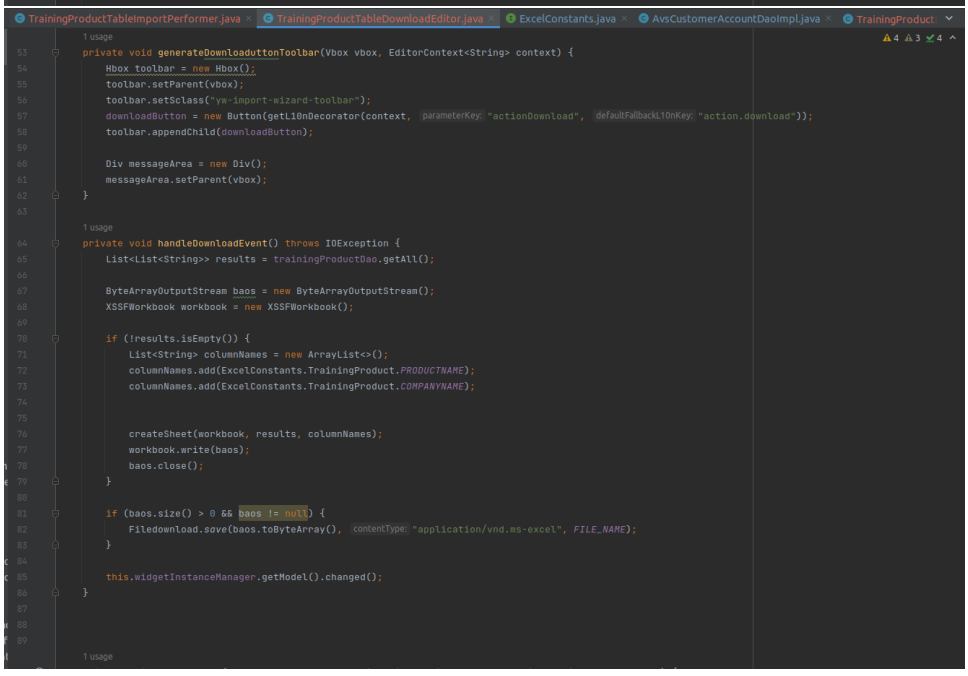
```xml
<!--    Task 9 excel ImportPerformer beans-->

    <bean id="trainingProductTableImportPerformer" parent="abstractMatchableExcelOperationPerformer"
          class="com.avansas.core.excel.TrainingProductTableImportPerformer">
        <property name="headers">
            <list>
                <util:constant static-field="com.avansas.core.excel.ExcelConstants.TrainingProduct.PRODUCTNAME"/>
                <util:constant static-field="com.avansas.core.excel.ExcelConstants.TrainingProduct.COMPANYNAME"/>
            </list>
        </property>
    </bean>

    <bean id="trainingProductTableImportPerformerOperation" class="com.avansas.core.excel.MatchingExcelOperationHandler"
          parent="matchingExcelOperationHandler">
        <property name="performers">
            <list>
                <ref bean="trainingProductTableImportPerformer"/>
            </list>
        </property>
    </bean>
```

3- Download için yine referans sınıfı örnek olarak alarak TrainingProductTableDownloadEditor sınıfı oluşturuyoruz.


TrainingProductTableImportPerformer.java
TrainingProductTableDownloadEditor.java
ExcelConstants.java
AvsCustomerAccountDaoImpl.java
TrainingProduct
1 usage
public class TrainingProductTableDownloadEditor extends AbstractCockpitEditorRenderer<String> {
protected WidgetInstanceManager widgetInstanceManager;
3 usages
private Button downloadButton;
1 usage
@Resource
private TrainingProductDao trainingProductDao;
1 usage
private static final String TRAINING_PRODUCT_TABLE = "Training Product Table";
1 usage
private static final String FILE_NAME = "Training Product Table";
@Override
public void render(Component component, EditorContext<String> editorContext, EditorListener<String> editorListener) {
Validate.notNull(message: "All parameters are mandatory", new Object[]{component, editorContext, editorListener});
this.widgetInstanceManager = ((Editor) component).getWidgetInstanceManager();
Div containerDiv = new Div();
containerDiv.setParent(component);
Vbox vbox = new Vbox();
vbox.setParent(containerDiv);
generateDownloaduttonToolbar(vbox, editorContext);
this.downloadButton.addEventListener(evtnm: "onClick", (event) -> this.handleDownloadEvent());
}
1 usage
private void generateDownloaduttonToolbar(Vbox vbox, EditorContext<String> context) {
Hbox toolbar = new Hbox();
toolbar.setParent(vbox);
toolbar.setSclass("ym-import-wizard-toolbar");
downloadButton = new Button(getL10nDecorator(context, parameterKey: "actionDownload", defaultFallbackL10nKey: "action.download"));
toolbar.appendChild(downloadButton);


TrainingProductTableImportPerformer.java
TrainingProductTableDownloadEditor.java
ExcelConstants.java
AvsCustomerAccountDaoImpl.java
TrainingProduct
1 usage
private void generateDownloaduttonToolbar(Vbox vbox, EditorContext<String> context) {
Hbox toolbar = new Hbox();
toolbar.setParent(vbox);
toolbar.setSclass("ym-import-wizard-toolbar");
downloadButton = new Button(getL10nDecorator(context, parameterKey: "actionDownload", defaultFallbackL10nKey: "action.download"));
toolbar.appendChild(downloadButton);
Div messageArea = new Div();
messageArea.setParent(vbox);
}
1 usage
private void handleDownloadEvent() throws IOException {
List<List<String>> results = trainingProductDao.getAll();
ByteArrayOutputStream baos = new ByteArrayOutputStream();
XSSFWorkbook workbook = new XSSFWorkbook();
if (!results.isEmpty()) {
List<String> columnNames = new ArrayList<>();
columnNames.add(ExcelConstants.TrainingProduct.PRODUCTNAME);
columnNames.add(ExcelConstants.TrainingProduct.COMPANYNAME);
createSheet(workbook, results, columnNames);
workbook.write(baos);
baos.close();
}
if (baos.size() > 0 && baos != null) {
Filedownload.save(baos.toByteArray(), contentType: "application/vnd.ms-excel", FILE_NAME);
}
this.widgetInstanceManager.getModel().changed();
}
1 usage

```java
1 usage
private void createSheet(XSSFWorkbook workbook, List<List<String>> results, List<String> columnNames) {
    XSSFSheet sheet = workbook.createSheet(TRAINING_PRODUCT_TABLE);
    writeDataToSheet(sheet, columnNames, results);
}

1 usage
private void writeDataToSheet(XSSFSheet sheet, List<String> columnNames, List<List<String>> results) {
    HashMap<Integer, List<String>> data = new HashMap<>();
    data.put(1, columnNames);
    AtomicInteger atomicInteger = new AtomicInteger( initialValue: 1);

    results.stream().forEach(result -> {
        data.put(atomicInteger.incrementAndGet(), result);
    });

    //Iterate over data and write to sheet
    Set<Integer> keyset = data.keySet();
    int rownum = 0;
    for (Integer key : keyset) {
        XSSFRow row = sheet.createRow(rownum++);
        List<String> objArr = data.get(key);
        int cellnum = 0;
        for (String obj : objArr) {
            org.apache.poi.ss.usermodel.Cell cell = row.createCell(cellnum++);
            cell.setCellValue(obj);
        }
    }
}
```

4- TrainingProduct export formu oluşturuyoruz indirirken url modelimizdir .

```java
package com.avansas.backoffice.widgets;


public class TrainingProductExportForm {

    private String url;


    public String getUrl() {
        return url;
    }

    public void setUrl(String url) {
        this.url = url;
    }

}
```

5-avansasbackoffice-backofficespring.xml'de download wizard'ımızı oluturuyoruz

```xml
<context type="hmc_treenode_training_product_table_download_wizard" component="create-wizard">
    <wz:flow id="TrainingProductTableDownloadWizard" title="excel.training.product.export.title">
        <wz:prepare id="impExImportPrepare">
            <wz:initialize property="TrainingProductExportForm"
                           type="com.avansas.backoffice.widgets.TrainingProductExportForm"/>
        </wz:prepare>
        <wz:step id="step1" label="excel.export.prepare.label" sublabel="impex.export.prepare.sublabel">
            <wz:content id="step1.content">
                <wz:property-list root="TrainingProductExportForm">
                    <wz:property qualifier="url" editor="com.avansas.editor.trainingproductexport"/>
                </wz:property-list>
            </wz:content>
            <wz:navigation id="step1.navigation">
                <wz:done/>
            </wz:navigation>
        </wz:step>
    </wz:flow>
</context>
```

```xml
<context type="hmc_treenode_traini                                    onent="configurableFlowDynamicForms">
    <df:dynamicForms modelProperty                                    ris.com/cockpitng/component/dynamicForms">
        <df:attribute id="dynaSldForData" triggeredOn="distributedImpex" qualifier="sldForData"
                      disabledIf="distributedImpex==false" visibleIf="distributedImpex==true"/>
        <df:attribute id="dynaNodeGroup" triggeredOn="distributedImpex" qualifier="nodeGroup"
                      disabledIf="distributedImpex==false" visibleIf="distributedImpex==true"/>
        <df:attribute id="dynaLogLevel" triggeredOn="distributedImpex" qualifier="logLevel"
                      disabledIf="distributedImpex==false" visibleIf="distributedImpex==true"/>
        <df:attribute id="dynaProcessNumber" triggeredOn="distributedImpex" qualifier="processNumber"
                      disabledIf="distributedImpex==true" visibleIf="distributedImpex==false"/>
        <df:attribute id="dynaErrorMode" triggeredOn="distributedImpex" qualifier="errorMode"
                      disabledIf="distributedImpex==true" visibleIf="distributedImpex==false"/>
        <df:attribute id="dynaLogAuditTrail" triggeredOn="distributedImpex" qualifier="logAuditTrail"
                      disabledIf="distributedImpex==true" visibleIf="distributedImpex==false"/>
    </df:dynamicForms>
</context>
```

```
<context component="explorer-tree" merge-by="module">
    <explorer-tree:explorer-tree xmlns:explorer-tree="http://www.hybris.com/cockpitng/config/explorertree">
        <explorer-tree:navigation-node id="hmc_treenode_system">
            <explorer-tree:navigation-node id="hmc_treenode_systemtools">
                <explorer-tree:navigation-node id="hmc_treenode_excel_export_wizard"/>
                <explorer-tree:navigation-node id="hmc_treenode_cutoff_download_wizard"/>
                <explorer-tree:navigation-node id="hmc_treenode_w2p_download_wizard"/>
                <explorer-tree:navigation-node id="hmc_treenode_reference_distance_download_wizard"/>
                <explorer-tree:navigation-node id="hmc_treenode_training_product_table_download_wizard"/>
                <explorer-tree:navigation-node id="hmc_treenode_excel_operation_wizard"/>
                <explorer-tree:navigation-node id="hmc_treenode_landing_table_excel_operation_wizard"/>
                <explorer-tree:navigation-node id="hmc_treenode_bundle_excel_operation_wizard"/>
                <explorer-tree:navigation-node id="hmc_treenode_new_bundle_excel_operation_wizard"/>
                <explorer-tree:navigation-node id="hmc_treenode_cdn_purge_wizard"/>
            </explorer-tree:navigation-node>
        </explorer-tree:navigation-node>
    </explorer-tree:explorer-tree>
</context>
```

Explorer-tree'de de belirliyoruz(daha 2 yerde belirli)

6-widgets'te ve connections'ları yazıyoruz backoffice için :

```
<!--Training widget-->
    <widget id="training_product_download_conditionevaluator" widgetDefinitionId="com.hybris.cockpitng.conditionevaluator"
            slotId="cockpitWidgetChildrenInvisible"
            template="false">
        <widget id="training_product_download_propertyextractor"
                widgetDefinitionId="com.hybris.cockpitng.widgets.common.propextractor"
                slotId="cockpitWidgetChildrenInvisible"
                template="false">
            <setting key="expression" type="String" value="id"></setting>
            <setting key="socketDataType_$T" type="String" value="java.lang.String"></setting>
            <setting key="widgetStyleAttribute" type="String" value=""></setting>
            <setting key="widgetStyleClass" type="String" value=""></setting>
            <virtual-sockets/>
        </widget>
        <setting key="expression" type="String" value="#root.id EQ 'hmc_treenode_training_product_table_download_wizard'"></setting>
        <setting key="socketDataType_$T" type="String"
                 value="com.hybris.backoffice.navigation.NavigationNode"></setting>
        <setting key="widgetStyleAttribute" type="String" value=""></setting>
        <setting key="widgetStyleClass" type="String" value=""></setting>
        <virtual-sockets/>
    </widget>
```

```
<widget-connection sourceWidgetId="STUB_com.hybris.cockpitng.editor.bundleoperation" outputId="referenceEditorOutput" targetWidgetId="configurableFlow" input
<widget-connection sourceWidgetId="configurableFlow" outputId="wizardResult" targetWidgetId="STUB_com.hybris.cockpitng.editor.bundleoperation" inputId="re
<widget-connection sourceWidgetId="STUB_com.hybris.cockpitng.editor.bundleoperation" outputId="referenceSelected" targetWidgetId="collectionEditorAreaGroup"
<widget-connection sourceWidgetId="STUB_com.hybris.cockpitng.editor.bundleoperation" outputId="referenceSearchCtx" targetWidgetId="referenceadvancedsearchg

widgets>
```

7-Download için editor tanımlamasını oluşturuyorum düğümün iyi gözükmesi için .

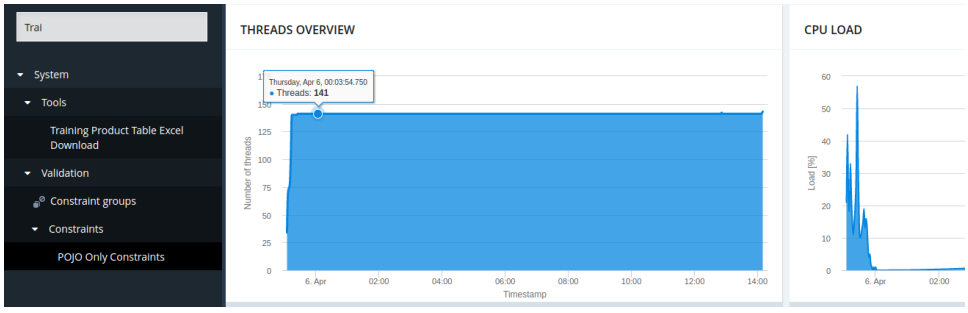
```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<editor-definition id="com.avansas.editor.trainingproductexport"
                   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                   xsi:noNamespaceSchemaLocation="http://www.hybris.com/schema/cockpitng/editor-definition.xsd">

    <name>Training Product Download Editor</name>
    <description>Download excel file of all Training products </description>
    <author>Barkai</author>
    <version>0.1</version>
    <type>java.lang.String</type>

    <editorClassName>com.avansas.backoffice.widgets.TrainingProductTableDownloadEditor</editorClassName>
</editor-definition>
```

8- Test :

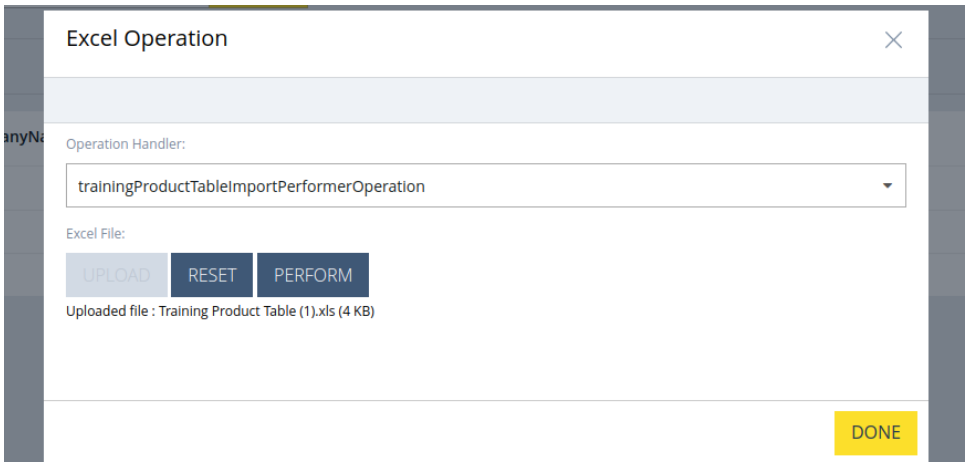


Tools'un altına excel download'umuz geldi başararıyla .

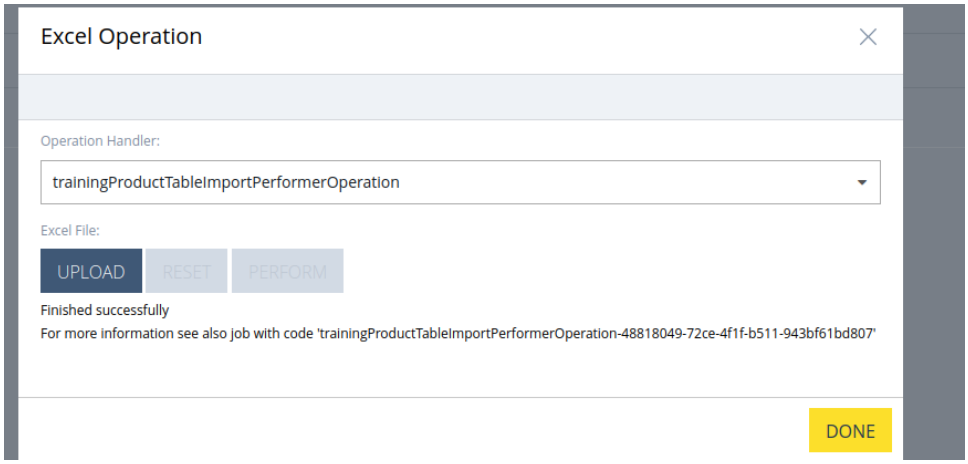
	A	B	C
1	Product Nam	Company Name	
2	moise	samsun	
3	mackbook	apple	
4			
5			

	A	B	C
1	Product Nam	Company Name	
2	moise	samsun	
3	mackbook	apple	
4	test_urun	test_sirket	
5			

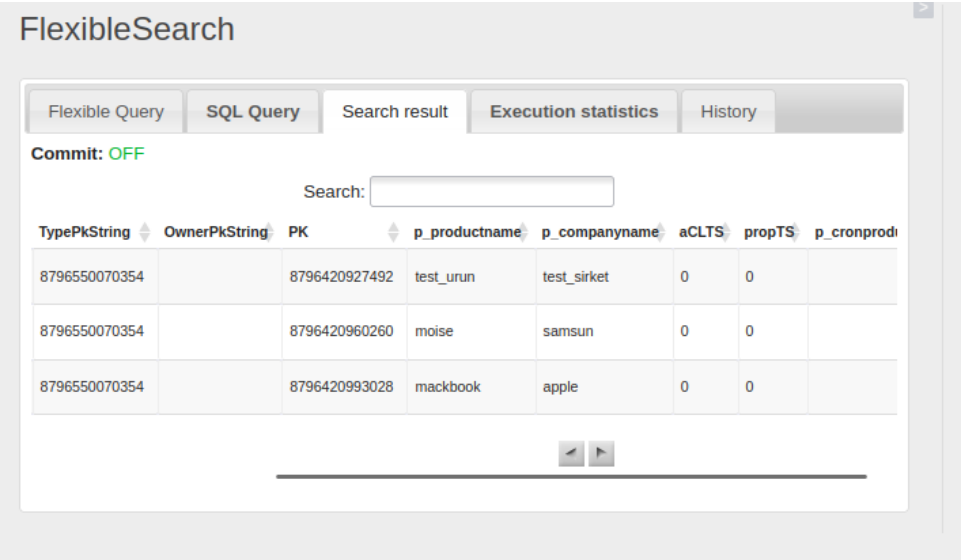
ProductName ve companyName ekleyip import edip bakazağız veri tanına save edecek mi ?



Değiştirdiğimiz excel file'mizi upload ediyoruz.



Başarıyla import edildi .



Test başarıyla gerçekleşti.